Simulating OS Personalities in NS3 Nodes

Diana Darabi
Polytechnic University of Puerto Rico
Mentor: Peter Barnes, PhD, Lawrence Livermore National Laboratory

Introduction

As our dependence on technology and need to interconnect information resources grows, so does the need for more realistic network simulations. In this poster we will be looking at the NS-3 network simulator. A limitation in NS-3 is that the nodes are generic which does not allow for realistic network simulations. We looked at the honeyd personality engine to be able to generate NS-3 nodes with realistic behaviours.

NS-3

- NS-3 is an open-source discrete-event network simulator
- Used mainly for educational and research purposes.
- It has multiple levels of abstraction and allows for protocol design and prototyping.
- NS-3 has generic nodes that do not represent any particular OS behaviour.

Honeyd

- Honeyd is a framework for simulating low-interaction virtual honeypots
- Using Nmap fingerprints, honeyd is able to emulate the network stack behavior of the personality assigned to each virtual honeypot.

Honeyd Personality Engine

- The Honeyd personality engine uses the Nmap fingerprint database to emulate OS personalities on its nodes.
- It then generates the response packets to simulate that of the OS personality assigned to the node.
- Changes IP packet information, sequence numbers, TCP packet headers, TCP flags, TCP window size, UDP packet headers, and ICMP responses accordingly.

Setting Up the OS Personality

```
### Windows NT4 web server
create windows
set windows personality "Windows NT 4.0 Server SP5-SP6"
add windows tcp port 80 "perl scripts/iis-0.95/iisemul8.pl"
add windows tcp port 139 open
add windows tcp port 137 open
add windows udp port 137 open
add windows udp port 135 open
set windows default tcp action reset
set windows default udp action reset

bind 10.0.1.51 windows
```

Nmap

- "Network Mapper“ - An open-source network discovery and security auditing tool
- Nmap sends a series of up to 16 TCP, UDP and ICMP probes to known open and closed ports and analyses the response packets to discover the OS running on the nodes in a network.
- Honeyd uses a subset of the fields in the Nmap OS fingerprint file.

Level of Detail of an Nmap OS Personality

There are currently 138 OS families in the Nmap OS fingerprint database.

Example of Fingerprint and Class lines providing a detailed description of the node personality:

```
Fingerprint Linux 2.0.33
Class Linux | Linux | 2.0.X | general purpose
```

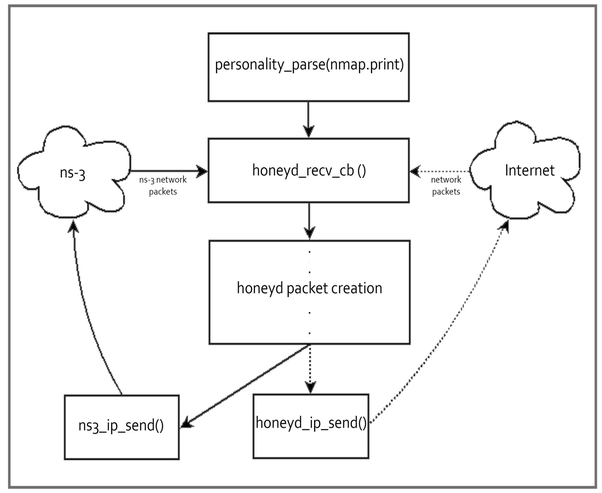
OS Classification

Class Categories	Description	Example
Vendor	The company that makes the device or OS	Linksys, Microsoft, Apple
OS Families	The OS Platform. Embedded is used for those devices with an undisclosed OS.	Windows, Linux, IOS
OS Generation	A more detailed description of the OS	Linux: 2.4.X, Windows: 2000, XP
Device type	A broad classification for the hardware.	general purpose (OS), router, printer

Nmap Fingerprint Fields

Fingerprint Item	Description
Fingerprint	A token to let Nmap know to load a new fingerprint, after the token is an OS description which includes the vendor, product name and version number.
Class	Device and OS Classification. Includes vendor, OS family, OS generation, and device type.
CPE	Common Platform Enumeration is a standardized way to name software applications, operating systems, and hardware platforms
SEQ	Contains results based on sequence analysis of the probes
OPS	Contains TCP options received in probe responses
WIN	Contains window sizes for prove responses
ECN	TCP explicit congestion notification.
T1	TCP probe packet with the SYN flag set sent to an open port
T2	TCP probe packet that sends a null packet with the IP DF bit set and a window field of 128 to an open port.
T3	TCP probe packet with the SYN, FIN, PSH, and URG flags set sent to an open port
T4	TCP probe packet with the ACK flag set, with IP DF and a window field of 1024 is sent to an open port
T5	TCP probe packet with the SYN flag set, without IP DF and with a window field of 31337 is sent to a closed port
T6	TCP probe packet with the ACK flag set, with IP DF and a window field of 32768 is sent to a closed port
T7	TCP probe packet with the FIN, PSH, and URG flags set, IP DF field not set, and a window field of 65535 is sent to a closed port
UI	UPD packet sent to a closed port. IP ID value is set to 0x1042 and the data field is set to the character ‘C’, repeated 300 times. If the port has no firewall in place and it is actually closed an ICMP port unreachable message is expected back from this probe.
IE	ICMP echo. In this test two ICMP echo requests are sent to the target immediately after the TCP sequence probes in order to obtain valid results of the shared IP ID sequence number test.

Implementation Sketch



Using honeyd to give ns-3 nodes personality.

Functions

personality_parse() - parses through the “nmap.prints” and creates the personality structure.
honeyd_recv_cb() - receives and processes network packets.
honeyd_ip_send() - sends the response packet that has already been created by honeyd based on the OS personality of the node.
ns3_ip_send() - takes the place of honeyd_ip_send() to process the packet through an ns-3 node.

Conclusion/Future Work

Based on the Nmap OS fingerprints, honeyd is able to simulate an OS personality that includes the specific vendor, OS family, OS generation, and device type and respond to packets based on the OS personality. There are currently 4,123 different OS fingerprints in the Nmap database with 138 different OS families that honeyd is able to take advantage of. The low-interaction personality engine would give NS-3 nodes the capability of responding to certain probes in a realistic fashion.

As outlined in our implementation sketch, we plan to make an NS-3 app based on the Honeyd personality engine to be able to assign OS personalities to NS-3 nodes.

References

Grimes, Roger A., Honeypots for Windows, Apress, 2005.

Lyon, Gordon “Fyodor”, Nmap Network Scanning, Nmap Project, 2009.

NS-3 Tutorial, Release ns-3.17, ns-3 project, May 2013. http://www.nsnam.org/docs/tutorial/html/index.html

Provos, Niels, *A Virtual Honeypot Framework*, www.citi.umich.edu/u/provos/papers/honeyd.pdf,

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

IM Review and Release #LLNL-POST-641946